Express Mail Label
No. EL 849003485 US

Attorney Docket No.
14406US03

APPENDIX F

Title : Radio Frequency Local Area Network

Inventors: Meier, et al.

Attorney Docket No. 14406US03

MAC LAYER SPECIFICATION

Introduction.

This document, in conjunction with the SST network frame format specification, defines the hop-to-hop link protocol used on RS485 LAN links, spread spectrum radio links and ethernet links.

Functional Requirements and Capabilities.

The MAC layer:

- accepts frames from the bridging layer and passes frames to the physical layer for transmission.
- appends MAC layer framing bytes and CCITT-16 FCS bytes to transmitted frames.
- removes MAC layer framing bytes and FCS bytes from received frames.
- verifies the FCS bytes in received frames.
- filters out frames which do not belong to the SST network of the local device.
- filters out packets which are not directed to the local device.
- forwards packets to the bridging layer which are directly addressed to the local device, or are broadcast or multicast to the local device.
- regulates access to the communications channel on RS485 links and spread spectrum radio links.
- schedules lost unicast frames for retransmission.
- detects and discards duplicate back-to-back unicast MAC level data frames.

- provides device-to-device flow control.
- transparently fragments and reassembles bridging layer packets, which exceed the maximum MAC frame size.
- maintains and provides diagnostic statistics for higher layers.

Terminology and Definitions.

frame - MAC layer protocol data unit.

packet - bridge layer protocol data unit.

A **channel access algorithm** is used to regulate access to a communications channel. The implementation of a channel access algorithm is dependent on the link type:

A **p-persistent CSMA/CA** (carrier sense multiple access with collision avoidance) protocol is used to gain access to an **RS485 LAN**. The collision avoidance scheme gives channel access priority to the recipient of a unicast frame.

On lightly loaded spread spectrum radio links, a **non-persistent CSMA** algorithm is used to gain access to the communications channel.

On moderately to heavily loaded spread spectrum radio links, an **LBT/BP (listen-before-talk with busy pulse)** algorithm is used to gain access to the channel and minimize the effect of hidden nodes.

A **polling protocol** is used to restrict contention to **request-for-poll (RFP)** frames, thus minimizing contention for data frames.

A **CSMA slot** is defined as follows: Assume that station A has determined that the channel is idle at time 0 and immediately initiates a transmission. Then a CSMA slot is equal to the worst-case time, t, at which another station, B, is able to determine that the transmission from A is in progress. The actual components of the CSMA slot time are <busy sense time> + <turnaround time> + <processing delay>, where each of the components represents a worst-case value. The **busy sense time** is the time required to detect a frame in progress. The **turnaround time** is the time required by a half-duplex transceiver to switch from a receive state to a transmit state. The processing delay includes the hardware/software processing time required to initiate a transmission and MAC frame processing time.

An **LBT slot** is defined as the total time required to transmit an RFP frame plus the time required by the receiver to begin transmitting the response.

A **CA (collision avoidance) sequence** of frames consists of one or more frames which are sent following a single execution of the channel access algorithm.

An **interframe gap** is defined as the minimum idle time allowed between frames which are considered as a single CA sequence.

An **interpoll gap** is defined as the maximum time allowed between poll frames which belong to a single CA sequence.

A **return priority mechanism** allows frames which are transmitted between a pair of nodes to be grouped into a single CA sequence which is initiated with one execution of the channel access algorithm.

CSMA idle time is the minimum time that a potential transmitter must sense an idle channel before assuming the channel is idle. The CSMA idle time is greater than the interframe gap plus the CSMA slot size.

LBT idle time is the minimum time that a potential transmitter must sense an idle radio channel with hidden nodes before assuming the channel is idle. The LBT idle time is greater than the interpoll gap time plus the CSMA slot size.

A **bracket** of frames is defined as a group of one or more MAC-level frames which are associated with a single bridging layer unit of data.

MAC-level frames are categorized as either **request** or **poll** frames:

A **DATA frame** is a MAC-level **request** frame which is used to send higher-layer data to a receiver.

An **EOD (end-of-data) frame** is a MAC-level **request** frame which is sent as the last data frame in a bracket of one or more data frames. Note that a bracket of data frames may consist of a single EOD frame.

An **RFP (request-for-poll) frame** is a MAC-level **request** frame which is used to request polling from a receiver and to determine the SEQ state of the receiver.

An **ENQ (enquiry) frame** is a MAC-level **request** frame which is used to determine the SEQ state of a receiver and to determine if a node is within range.

A **POLL frame** is a MAC-level **poll** frame which is used to obtain a data frame from another node and to return the current SEQ state.

A **WFP (wait-for-poll) frame** is a MAC-level **poll** frame which is used to inform a requesting node that it is scheduled to be polled later and to return the current SEQ state.

A **CLEAR frame** is a MAC-level **poll** frame which is used to inform all listening nodes that the last frame in a bracket of frames has been received and to return a defined SEQ state.

A **REJECT frame** is a MAC-level **poll** frame which is used to return an undefined SEQ state or to indicate that a received request frame was invalid.

MAC control byte bit definitions.

Request or poll frame control byte bit definitions.

R/P (request/poll) bit.

The R/P bit is used to distinguish MAC layer request and poll frames. If the R/P bit is set OFF the frame is a request frame. If the R/P bit is set ON the frame is a poll frame.

SEQ bit.

The SEQ bit is used to sequence MAC layer data frames, modula 2. The SEQ field is used to detect and discard duplicate packets. A state machine which illustrates the use of the SEQ bit and the response ACC bit is shown below.

LAN ID bits.

The MAC frame belongs to the spanning tree specified by the LAN ID bits. The MAC entity discards frames which belong to spanning trees which are not in its LAN_ID_list. Note that LAN_ID_list is a parameter of the MAC_enable call.

Request control byte bit definitions.

DATA bit.

The DATA bit is used to distinguish control request frames from data request frames.

MORE bit.

In control request frames the MORE bit is used to distinguish RFP frames from ENQ frames.

In data request frames, the MORE bit is used to distinguish between DATA frames and EOD frames. The last frame sent in a bracket of data frames is always an EOD frame.

PRIORITY bit.

The PRIORITY bit indicates the priority of a higher layer message and is set as specified by the bridging layer. in the MAC_send call. The receiver simply passes the priority to the bridging layer. The PRIORITY bit value is the same for all frames which are associated with a bracket of frames

Poll control byte bit definitions.

MORE bit.

The MORE bit is used to distinguish POLL frames from CLEAR frames.*

WAIT bit.

The WAIT bit is used to distinguish POLL frames from WFP frames.* The receiver of a request frame can return a poll frame with the WAIT bit set ON in the associated poll frame to put the requesting node in a quiet state for WFP_TIMEOUT seconds. The requesting node must refrain from transmitting unicast frames to the receiver until the quiet period expires or a POLL frame is received from the receiver.

*A REJECT frame is specified by setting the MORE bit OFF and the WAIT bit ON.

Bridging Layer Interface Specification.

Each node in the network has a single bridging entity which invokes a MAC entity per port to send and receive messages on the port.

MAC layer services are provided with the following routines:

MAC_enable(port, LAN_ID_list)

MAC_set_address(port, net_address)

MAC_send(port, dest_net_address, buffer, control_flags, [mailbox], [queue])

control_flags bits (7-0)

bit 7	priority	0 = normal, 1 = high
bit 6	reserved	must be zero
bit 5	reserved	must be zero
bit 4	reserved	must be zero
bit 3	p_flag	1 = p-persistent
bit 2	reserved	must be zero
bit 1-0	LAN ID	(spanning tree ID)

length=**MAC_accept**(port, buffer, wait)

MAC_stop(port)

MAC_start(port)

MAC_disable(port)

MAC_enquiry(port, dest_net_address)

MAC_diagnostic(port, . . .)

Initially, the MAC entity attached to a port is in a DISABLED/OFF state. The bridging layer enables a MAC entity on a port by calling MAC_enable(port, LAN_ID_list), where LAN_ID_list defines the spanning trees to which the node can belong. MAC_enable changes the MAC entity state to ENABLED/ON.

The MAC entity uses a default multicast address consisting of the node type and a node identifier of all 1's, until the bridging layer assigns a specific network address to the MAC entity. The MAC_set_address call is provided for this purpose.

The bridging layer accepts messages from the MAC entity by issuing a MAC_accept call. The returned buffer includes the MAC header, but does not including media framing and CRC characters. The wait parameter can be used to suspend the caller for some length of time or until a message is received. The MAC entity must be capable of queueing messages until they are accepted by the bridging layer.

The bridging layer requests the MAC entity to transmit a bridging layer packet by issuing a call to MAC_send. Packets are grouped into a set of one or more MAC layer frames which, together, consitute a bracket. On radio ports, if the size of a bridging layer packet exceeds the maximum MAC frame length, then the packet is fragmented. A bracket normally contains a single data (EOD) frame on wired links. The MAC entity prefixes a MAC header to the beginning of each frame in a bracket before transmitting each frame. The MAC layer is also responsible for providing media framing, which includes a link-type-dependent synchronization preamble, start-of-frame delimiter, end-of-frame delimiter, and CRC-CCITT frame check sequence bytes for each frame. The **control_flags** parameter in the MAC_send call is used to 1) set the priority bit in the MAC header (priority), 2) to indicate if the buffer is being sent in response to a multicast bridging layer packet (p_flag), and 3) to set the LAN ID field in

the MAC header. The optional mailbox and queue parameters are mutually exclusive and are used for asynchronous calls.

The maximum size of a buffer passed to the MAC layer for transmission is MAX_PKT_SIZE.

The bridging layer can disable the MAC receiver by calling MAC_stop. The MAC entity is in an ENABLED/OFF state after a call to MAC_stop is issued. The bridging layer forces the MAC entity back into the ENABLED/ON state by calling MAC_send or MAC_start.

The bridging layer can disable the MAC entity and force it to the DISABLED/OFF state by calling MAC_disable.

MAC_enquiry can be use to determine if a destination node is within range.

MAC_diagnostic is used to retrieve diagnostic statistics from the MAC layer.

Link Interface Specification.

Frame/packet filtering.

When the MAC entity is in an ENABLED/ON state it is continuously listening on its assigned port. The MAC entity receives all MAC layer frames. Frames which do not pass a CRC-CCITT check are invalid and are discarded. Valid data frames are reassembled into a complete packet which is posted to the bridging entity if:

1) The LAN ID in the MAC header is among those contained in the LAN ID list passed to the MAC entity in the MAC_enable call, and

2) The destination address in the MAC header a) is equal to the network address of the local node, or b) is an acceptable multicast or broadcast address.

The high-order multicast bit is set ON in all multicast or broadcast frames. A multicast or broadcast frame is accepted if the node type specifies a group to which the local node belongs and either a) the node identifer is all 1's, or b) the node identifer is equal to the identifier of the local node. A response is never required when the multicast bit is set ON.

The default network address used when the MAC entity is first enabled consists of the multicast node type concatenated with a node identifier of all 1's. For example, the default address for a bridge is hexidecimal A7FF. The bridging layer is responsible for obtaining a network address and assigning it to the MAC entity on the port.

Channel access.

A return priority mechanism is used to group MAC layer request and poll frames into a single CA sequence. A channel access algorithm is executed to gain access to the channel before the first frame in a CA sequence is transmitted. All other frames in a CA sequence may be sent without executing the channel access algorithm. The idle time between frames which belong to a single CA sequence must be less than the maximum interframe gap time.

On wired links, the CSMA/CA algorithm forces nodes to detect an idle channel for a CSMA idle time which exceeds the interframe gap time before initiating a CA sequence.

On radio links "hidden nodes" can cause throughput to be significantly degraded on spread spectrum radio links. Under lightly loaded conditions, a CSMA channel access algorithm allows nodes to access the radio channel immediately after detecting an idle channel. Under moderate to heavily loaded conditions, the **LBT/BP** algorithm forces nodes to detect an idle radio channel for an **LBT idle time** which exceeds the **interpoll gap** time, before accessing the channel. By listening for longer than the interpoll gap time, a node will detect a conversation in progress, if both involved nodes are in range or only one node is in range and the other node is hidden. Limiting the time between frames in a CA sequence to a short fixed interval, essentially provides a **busy-pulse** signal which spans the coverage area of both nodes involved in a conversation.

A CA sequence of frames begins with the transmission of a request or poll frame, following an execution of the channel access algorithm. Possible successive frames in a CA sequence are:

1) Any poll frame sent in response to a unicast request frame.
2) A DATA or EOD frame sent in response to a POLL frame.
3) A bridge node can "piggyback" a second frame onto a transmitted broadcast, multicast, WFP, CLEAR, or REJECT frame, by transmitting the second frame within the interframe gap time.

Message Bracketing.

The size of packets which are passed to the MAC layer by the bridging layer must be less than, or equal to, MAX_PKT_SIZE, where MAX_PKT_SIZE specifies the total length of the packet, including bridging and data-link header characters.

Packets which are larger than MAX_FRAME_SIZE must be fragmented, by the MAC entity, to insure that the interpoll gap time is constant. The fragmented frames are transmitted as a **bracket** with the MORE bit set OFF in the last frame to mark the end of the bracket. Frames which belong to a single bracket are reassembled by the MAC entity in the receiver before the packet is posted to the bridging layer in the receiver. If the entire bracket is not received successfully, then all other frames in the bracket are discarded by the receiver. Note that the maximum number of data frames in a bracket is the ceiling of MAX_PKT_SIZE / MAX_FRAME_SIZE.

MAX_FRAME_SIZE does not include characters added at the MAC level. MAX_FRAME_SIZE on the 192K bps spread spectrum radio link is limited by the interpoll gap time.

On a wired links with low error rates, MAX_FRAME_SIZE is set so that a bracket is generally limited to a single **EOD frame.**

A bracket of frames may be transmitted in one or more CA sequences, where a channel access algorithm is used to gain access to the link for each CA sequence. A transmitter initiates the transmission of a bracket of frames by sending either an RFP frame or an EOD frame to a receiver. If a receiver is not busy, the receiver will respond to RFP and DATA frames with a POLL frame, which solicits the next DATA frame and implicitly acknowledges the previous frame. A receiver responds to an EOD frame with a CLEAR frame. If a receiver is busy or does not have a buffer, then the receiver may respond to RFP, DATA or EOD frames with a WFP frame.

Example transmit/receive sequences.

The following sequences illustrate typical transmission sequences, in the absence of errors, for sending a bracket of data frames from a transmitter to a receiver.

Sequence 1:

```
RFP    ---------->
       <---------- POLL
EOD    ---------->
       <---------- CLEAR
```

Sequence 2:

```
EOD    ---------->
       <---------- CLEAR
```

Sequence 3:

```
RFP    ---------->
       <---------- POLL
DATA   ---------->
          .
          .
       <---------- POLL
EOD    ---------->
       <---------- CLEAR
```

Sequence 4:

```
RFP    ---------->
       <---------- WFP
          (pause)
       <---------- POLL
DATA   ---------->
          .
          .
       <---------- POLL
EOD    ---------->
       <---------- CLEAR
```

Sequence 5:

```
ENQ    ---------->
       <---------- CLEAR
```

Sequences 3 through 4 can only occur on radio links. Note that, in sequence 2, an EOD frame is sent immediately, without sending a prior RFP frame. It is possible to skip the transmission of an RFP frame if the transmitter "remembers" the SEQ state of the receiver. This facility is limited to the transmission of short EOD frames on wired links. Sequence 5 can be used to determine the SEQ state of a receiver or to determine if a node is within range.

Recovery.

The node which initiates a bracket of frames (i.e. the transmitter) is responsible for recovery until the first POLL frame is received.

The receiver is responsible for polling the transmitter as soon as an RFP frame is received and assumes responsibility for recovery at that point.

It is possible for both the transmitter and receiver to be in contention to recover a lost frame (i.e. RFP or DATA) if the first POLL frame is lost. The contention is resolved with a random backoff algorithm.

If a CLEAR frame is lost and the polling node which sent the CLEAR frame is responsible for recovery, then the requesting node which initiated the bracket can not determine if the link was lost or the CLEAR frame was lost. The requesting node must send an ENQ frame to determine which case holds.

Transmit and Receive State Machine (SM) Specifications.

No state machine is required for multicast and broadcast frames. Multicast and broadcast frames can be transmitted whenever the channel is available. Received multicast or broadcast frames are simply discarded or posted to the bridging layer.

State Machines for Bracket Transmission.

Bracket Transmit State Machine.

State descriptions:

IDLE - The state machine is idle and is waiting for a bracket of frames to transmit.

READY - The state machine has a bracket of 1 or more frames to transmit and is waiting to acquire the channel.*

S_RFP - The state machine has sent an RFP frame and is waiting for a POLL frame.

S_DATA - The state machine has sent a DATA frame and is waiting for a POLL frame.

S_EOD - The state machine has sent an EOD frame after receiving a POLL frame and is waiting for a CLEAR frame.

RDY_WAIT - The state machine has received a WFP frame and is waiting for a POLL frame (or timeout).

The following states only apply to transmissions on a wired link which are not initiated with a request for polling.

READY2 - The state machine has a single short frame to transmit, is waiting to acquire a wired link, and the SEQ state of the receiver is known.

S_EOD_2 - The state machine has sent an unsolicited EOD frame is waiting for a CLEAR frame.

* There is an automatic and immediate transition from the READY state to the READY2 state if all of the following conditions are true: 1) the communciations channel is a wired link, 2) the SEQ state of the receiver is known, and 3) the bracket to transmit consists of a single EOD frame which is less than MAX_SHORT_FRAME_SIZE in length.

Timers:

A RSP_TIMEOUT receive timer is started when a) an RFP frame is transmitted, 2) an ENQ frame is transmitted, and 3) on wired links, when an EOD frame is sent without first sending an RFP frame. The timeout value is larger than interframe gap time plus the time required to transmit a POLL or CLEAR frame. If the RSP_TIMEOUT timer expires before an expected response is received, a retry counter is incremented and the request frame is retransmitted, if the retry count has not been exceeded.

A POLL_TIMEOUT receive timer is started whenever a DATA or EOD frame is transmitted following an RFP frame. The timeout value is larger than the time required for the maximum number of poll retry attempts. The MAC layer returns an error to the bridging layer if this timer expires before an expected poll frame is received. Note that the receiver is responsible for recovery when this timer is running.

A WFP_TIMEOUT timer is started whenever a WFP frame is received. The RDY_WAIT state ends when this timer expires or a POLL frame is received.

The state machine must maintain a "current pointer" variable which points to the current frame, in a bracket of frames, to be transmitted. The current pointer is advanced if, and only if, a POLL for the next frame in the bracket is received. If more than one transition is specified when a POLL frame is received, the state of the current pointer determines which transition should be taken.

Retransmission logic and SEQ state maintainence are specified in the section which describes state machines for frame SEQ control.

The first state table below specifies transitions for bracket transmissions which are initiated with a polling request. The second table specifies transitions for non-polled frame transmission (i.e. a single short EOD frame on a wired link). Note that there are transitions between the tables.

state	event	action	next state
IDLE	A bracket of frames is passed to the state machine	Reset retry count; execute channel access algorithm	READY
READY	Channel acquired	Increment retry count; send RFP frame; start RSP_TIMEOUT receive timer	S_RFP
S_RFP	RSP_TIMEOUT timer expires and max. retry count exceeded	Return error	IDLE
	RSP_TIMEOUT timer expires	Execute channel access algorithm	READY
	POLL received	Send current DATA frame; start POLL_TIMEOUT receive timer	S_DATA
	POLL received	Send current EOD frame; start POLL_TIMEOUT receive timer	S_EOD
	WFP received	Start WFP_TIMEOUT timer	RDY_WAIT
S_DATA	POLL_TIMEOUT timer expires	Return error	IDLE
	POLL received	Advance current pointer if frame was accepted; send current DATA frame; start POLL_TIMEOUT receive timer	S_DATA
	POLL received	Advance current pointer if frame was accepted; reset retry count; send current EOD frame; start POLL_TIMEOUT receive timer	S_EOD
S_EOD	POLL_TIMEOUT timer expires	Reset retry count; execute channel access algorithm	RDY_ENQ
	POLL received	Retransmit EOD frame; start POLL_TIMEOUT receive timer	S_EOD
	CLEAR received; EOD frame not accepted	Return error (invalid transition)	IDLE
	CLEAR received; EOD frame accepted	Return good	IDLE
RDY_ENQ	Channel acquired	Increment retry count; send ENQ; start RSP_TIMEOUT timer	S_ENQ
S_ENQ	RSP_TIMEOUT timer expires and max. retry count exceeded	Return error	IDLE
	RSP_TIMEOUT timer expires	Execute channel access algorithm	RDY_ENQ
	CLEAR received; EOD frame not accepted	Return error (invalid transition)	IDLE
	CLEAR received; EOD frame accepted	Return good	IDLE
RDY_WAIT	WFP received	Reset WFP_TIMEOUT timer	RDY_WAIT
	POLL received	Send current DATA frame; start POLL_TIMEOUT timer	S_DATA
	POLL received	Send current EOD frame; start POLL_TIMEOUT timer	S_EOD
	WFP_TIMEOUT timer expires	Reset retry count; execute channel acess algorithm	READY

state	event	action	next state
READY›	Channel acquired	Increment retry count; send EOD frame with RESET on; start RSP_TIMEOUT receive timer	S_EOD2
S_EOD›	RSP_TIMEOUT timer expires and max. retry count exceeded	Return error	IDLE
	RSP_TIMEOUT timer expires	Execute channel access algorithm	READY›
	WFP received	Start WFP_TIMEOUT timer	RDY_WAIT
	CLEAR received; EOD frame not accepted	Return error (invalid transition)	IDLE
	CLEAR received; EOD frame accepted	Return good	IDLE

Bracket Receive State Machine.

The receive state machine assumes that invalid frames and frames not directed to the local node are discarded and do not affect state transitions. Multicast and broadcast frames are simply posted to the bridging entity, if a buffer is available, and do not affect state transitions.

State descriptions:

IDLE_LISTEN - The receiver is not receiving a bracket of frames.

BUSY - The receiver has sent a POLL frame and is waiting for the next frame in a bracket.

BUSY_WAIT - The receiver is waiting for a buffer to become free.

Timers:

A RSP_TIMEOUT receive timer is started when a POLL frame is transmitted. The timeout value is larger than interframe gap time plus the time required to transmit a DATA frame. If the RSP_TIMEOUT timer expires before an expected response is received, a retry counter is incremented and the POLL frame is retransmitted, if the retry count has not been exceeded.

The receiver must maintain a **poll_queue** which is a FIFO list of all terminals which have requested polling. (Note that a separate queue can be used for high priority requests.) Entries in the queue are aged so that they are discarded after WFP_TIMEOUT seconds. The entry at the front of the queue is considered **active**; all other entries in the queue are denoted as **queued**. Nodes which are not active or queued are denoted as **inactive**. Note that there is no active node in the IDLE_LISTEN state.

A SEQ state variable is cached for all nodes which have recently transmitted valid data frames. The SEQ state variable is updated as specified in the section which describes state machines for frame SEQcontrol.

Only one bracket may be in progress at a time. The receiver must reserve enough buffers for an entire bracket of frames before sending a POLL frame in response to an RFP frame. This ensures that the entire bracket will be accepted.

"**Flush**" deletes the entry for a node from the poll_queue, if it exists, and frees any buffers allocated to frames received from the node.

state	event	action	next state
IDLE_LISTEN	RFP received; buffers available	Send POLL; insert node in poll_queue; reset retry count; start RSP_TIMEOUT timer	BUSY
	RFP received; buffers not available	Send WFP; insert node in poll_queue	BUSY_WAIT
	EOD received; buffer available	Send CLEAR; post complete packet (if accepted); flush	IDLE_LISTEN
	EOD received; buffer not available	Send WFP; insert node in poll_queue	BUSY_WAIT
	ENQ received; entry for source node is in the SEQ state table	Send CLEAR	IDLE_LISTEN
	ENQ received; no entry for source node in the SEQ state table	Send REJECT	IDLE_LISTEN
	DATA received	Send REJECT	IDLE_LISTEN
BUSY	max. retries exceeded	Flush; delete SEQ state table entry	IDLE_LISTEN
	receive timeout	Increment retry count; execute channel access algorithm; acquire channel; resend POLL; start RSP_TIMEOUT timer	BUSY
	DATA received from active node	Reset retry count; send next POLL; start RSP_TIMEOUT timer	BUSY
	DATA received from inactive or queued node	Send REJECT	BUSY
	EOD received from active node; no queued nodes	Send CLEAR; reassemble and post complete packet; flush	IDLE_LISTEN
	EOD received from active node; 1 or more queued nodes	Send CLEAR; reassemble and post complete packet; flush; reset retry count; send POLL to node at front of poll_queue; start RSP_TIMEOUT timer	BUSY
	EOD received from inactive or queued node; buffer available	Send CLEAR; post complete packet (if accepted); flush	BUSY
	EOD received from inactive or queued node; buffer not available	Send WFP; insert node in poll_queue	BUSY
	ENQ received from inactive or queued node; entry for source node is in the SEQ state table	Send CLEAR; flush	BUSY
	ENQ received from inactive node or queued node; no entry for source node in the SEQ state table	Send REJECT; flush	BUSY

	ENQ received from acuve node; no queued nodes	Send REJECT ; flush; delete SEQ state table entry	IDLE_LISTEN
	ENQ received from acuve node; 1 or more queued nodes	Send REJECT; flush; delete SEQ state table entry; reset retry count; send POLL to node at head of poll_queue; start RSP_TIMEOUT timer	BUSY
	RFP received from inactive or queued node	Send WFP; insert node in poll_queue	BUSY
	RFP received from active node	Send WFP if poll_queue is not empty; flush; insert node in poll_queue: reset retry count; send POLL to node at head of poll_queue; start RSP_TIMEOUT timer	BUSY
BUSY_WAIT	buffer becomes available	Send POLL to node at head of poll_queue; start RSP_TIMEOUT timer	BUSY
	DATA received	Send REJECT	BUSY_WAIT
	EOD received	Send WFP; insert node in poll_queue	BUSY_WAIT
	ENQ received; entry for source node is in the SEQ state table	Send CLEAR; flush	BUSY_WAIT
	ENQ received; no entry for source node in the SEQ state table	Send REJECT; flush	BUSY_WAIT
	ENQ received from active node; no queued nodes	Send REJECT ; flush	IDLE_LISTEN
	RFP received	Send WFP; insert node in poll_queue	BUSY_WAIT
	buffer becomes available	Send POLL to active node; start RSP_TIMEOUT timer	BUSY

State Machines for Frame SEQ Control.

All unicast MAC data frames are sequenced with a 1-bit sequence number (SEQ). The sequence number is used to detect lost data frames and duplicate data frames.

The MAC entity in each node must maintain transmit and receive SEQ state tables for unicast messages. The receive SEQ state table contains an entry for each active MAC source node. The transmit SEQ state table contains an entry for each active destination node. Each entry consists of a 1-bit SEQ state variable and a network address. **Only unicast command frames affect state table entries.** As a rule, a receive table entry should be discarded before the counterpart transmit table entry (i.e. in another node) is discarded. Receive SEQ state table entries need only be kept long enough to ensure that retransmitted duplicates are not mistaken for valid frames. This implies that receive table entries must be kept for a period longer than the maximum transmit retry time for a single frame. An entry in the transmit SEQ state table can be kept until the space is required for a new entry. Strict state timing is not required because a transmitter, without a table entry for a potential receiver, can determine the state of a receiver, with an RFP frame, before transmitting data frames. Also, note that the MAC layer does not provide a reliable service. Lost frames and duplicates are detected by higher layers.

The SEQ state machine descriptions below specify how entries in the SEQ state tables are maintained. Note that "poll" is used to denote any poll frame (i.e. POLL, WFP, CLEAR, or REJECT) and "data" is used to denote any data frame (i.e. DATA or EOD). Specific frame types are always in upper case.

Receive SEQ Control State Machine.

SEQ State descriptions:

ACCEPT_0 - the receiver expects the next DATA or EOD packet to have a SEQ number of 0.

ACCEPT_1 - the receiver expects the next DATA or EOD packet to have a SEQ number of 1.

ACCEPT_ANY - the receiver will accept a DATA or EOD packet with a SEQ number of 0 or 1.

The MAC receiver caches receive SEQ state variables for active external source nodes. The variable can be set to one of three states listed above. A state of ACCEPT_ANY applies to all nodes which do not have entries in the receiver's SEQ state table. The receiver sets the SEQ bit in a poll frame to denote the next frame that the receiver expects.

State	Event	Action	Next State
ACCEPT_ANY	Receive RFP/ENQ	Return poll 0	ACCEPT_ANY
	Receive data 0	Accept frame and return poll 1	ACCEPT_1
	Receive data 1	Accept frame and return poll 0	ACCEPT_0
	Receive data 0, no buffer	Discard frame and return WFP 0	ACCEPT_0
	Receive data 1, no buffer	Discard frame and return WFP 1	ACCEPT_1
ACCEPT_0	Receive RFP/ENQ	Return poll 0	ACCEPT_0
	Receive data 0	Accept frame and return poll 1	ACCEPT_1
	Receive data 1	Discard frame and return poll 0	ACCEPT 0
	Receive data 0, no buffer	Discard frame and return WFP 0	ACCEPT_0
	State timeout	Delete state entry	ACCEPT_ANY
ACCEPT_1	Receive RFP/ENQ	Return poll 1	ACCEPT_1
	Receive data 1	Accept frame and return poll 0	ACCEPT_0
	Receive data 0	Discard frame and return poll 1	ACCEPT_1
	Receive data 1, no buffer	Discard frame and return WFP 1	ACCEPT_1
	State timeout	Delete state entry	ACCEPT_ANY

Transmit SEQ Control State Machine.

SEQ State descriptions:

SEND_0 - the transmitter sends the current data frame with a SEQ number of 0 and expects a POLL or CLEAR with a SEQ number of 1.

SEND_1 - the transmitter sends the current data frame with a SEQ number of 1 and expects a POLL or CLEAR with a SEQ number of 0.

UNKNOWN - the transmitter must send an RFP or ENQ frame to determine the SEQ state of the receiver.

The MAC transmitter maintains a transmit SEQ state variable per external node. The transmitter sets the SEQ field in DATA and EOD frames to the value of the transmit SEQ state variable. The state variable can be in one of the three states listed above. The UNKNOWN state applies to all nodes which do not have entries in the transmitter's state table. If the state is UNKNOWN, the transmitter must send an RFP or ENQ frame, to determine the SEQ state of the receiver, before sending a data frame.

The SEQ field in a poll frame denotes the next data frame expected. Each time a poll frame is received, the transmit SEQ state variable, associated with the source of the poll frame, is set to the value of the poll frame's SEQ field. A "current pointer" points to the current data frame in a bracket of data frames. The current pointer is advanced if the current data frame has been transmitted with a SEQ field value of 0(1) and a poll frame is received with a SEQ field value of 1(0).

On radio links, the SEQ state is set to UNKOWN as soon as the transmission of a bracket of frames is completed.

Network Constants.

WFP_TIMEOUT = 1 second, is the time that a node remains in a quiet state waiting for a POLL frame after a WFP frame is received.

MAX_PKT_SIZE = 800 bytes, is the maximum size of a bridging layer packet including bridging header characters.

R_MAX_FRAME_SIZE = 100 bytes, is the maximum size of a MAC layer frame on the spread spectrum radio link, not including MAC header and framing characters.

W_MAX_FRAME_SIZE = MAX_PKT_SIZE, is the maximum size of a MAC layer frame on the RS485 LAN, not including MAC header and framing characters.

W_MAX_SHORT_FRAME_SIZE = 200 bytes, is the maximum size of a MAC layer frame which can be transmitted without first sending a RFP frame on the RS485 LAN.

W_SLOT_SIZE = 50 microseconds, is the CSMA slot size for the RS485 LAN.

W_INTERFRAME_GAP = 200 microseconds, is the maximum interfame gap time for the RS485 LAN.

W_IDLE_TIME = W_INTERFRAME_GAP + W_SLOT_SIZE + 50 microseconds, is the CSMA idle time on the RS485 LAN.

R_SLOT_SIZE = 1000 microseconds, is the **LBT slot** size on the spread spectrum radio link.

R_INTERPOLL_GAP = 5000 microseconds, is the **interpoll gap** time on the spread spectrum radio link.

R_IDLE_TIME = R_INTERPOLL_GAP, is the **LBT idle time** on the spread spectrum radio link.

Channel access algorithms.

The CSMA/CA channel access algorithm used on the RS485 LAN differs from the LBT algorithm for radio links because of the hidden terminal factor in the radio network.

CSMA/CA summary:

The p-persistent CSMA/CA algorithm forces all nodes to detect an idle channel for one CSMA idle time unit, where a CSMA idle time unit is greater than the interframe gap time, before the channel is considered free. If a node initially detects a free channel, it can transmit immediately. If a node detects a busy channel, it listens to the channel until it becomes free. When the channel becomes free, at that point, time is divided into p CSMA slots. The node selects on of the p slots, say i, at random. If the channel is idle for the first i-1 slots, then the node will transmit in slot i. If the channel becomes busy in one of the first i-1 slots, the process is repeated. If an expected response is not received, a node will choose a number, i, between 1 and p, and will delay for i CSMA slots before re-executing the CSMA algorithm to retransmit. The number of backoff slots, p, is given as an increasing function of the number of missed responses and busy channel detections.

LBT/BP summary:

The LBT algorithm functions as a pure CSMA algorithm when the channel is lightly loaded. A channel is allowed to transmit as soon as an idle channel is detected. CSMA is never used for retransmissions. When the channel is moderately to heavily loaded, the LBT algorithm forces all nodes to detect an idle channel for at least one LBT idle time unit, where an LBT idle time unit is greater than the interpoll gap time, before the channel is considered free. If a node initially detects a free channel, it can transmit immediately. If a node detects a busy channel, it listens to the channel until it becomes free. When the channel becomes free, at that point, time is divided into p LBT slots. The node selects on of the p slots, say i, at random. If the channel is idle for the first i-1 slots, then the node will transmit in slot i. If the channel becomes busy in one of the first i-1 slots, the process is repeated. If an expected response is not received, a node will choose a number, i, between 1 and p, and will delay for i LBT slots before re-executing the LBT algorithm to retransmit. The number of backoff slots, p, is given as an increasing function of the number of missed responses and busy channel detections.

The CSMA/CA algorithm for the RS485 LAN, and the LBT/BP algorithm for spread spectrum radio links are both shown in pseudo-code below.

LBT/BP algorithm for a transmitter on the radio network.

```
BACKOFF_INIT     = 20;
R_RSP_TIMEOUT    = R_INTERPOLL_GAP;
MAX_TX_TRIES     = 20;
MAX_IDLE_TRIES   = 50;
```

rf, rg - functions which return a maximum backoff number based on the input parameter.

Wait for a MAC_send call.

```
if p_flag is non-zero then
begin
  select a random number, i, between 0 and BACKOFF_INIT;
  SLOT_OFFSET =i*R_SLOT_SIZE;
end
else
  SLOT_OFFSET=0 ;

TX_RETRIES=0;
IDLE_RETRIES=0;

while TX_RETRIES < MAX_TX_TRIES and IDLE_RETRIES < MAX_IDLE_TRIES and not OK do
begin
  OK=False;
  detect an idle channel for SLOT_OFFSET+R_IDLE_TIME time units;
  SLOT_OFFSET=0;
  if channel is idle then
  begin
    send_frame;
    if a return priority response is expected then
    begin
      wait for response or R_RSP_TIMEOUT timeout;
      if a valid response has been received then
        OK=True;
      else (assume a collision has occurred)
      begin
        TX_RETRIES=TX_RETRIES+1;
        select a random number, j, between 0 and rf(TX_RETRIES);
        SLOT_OFFSET=j*R_SLOT_SIZE;
      end
    end
  end
  else (the channel is not idle)
  begin
    wait until the channel is idle;
    IDLE_RETRIES=IDLE_RETRIES+1;
    select a random number, k, between 0 and rg(IDLE_RETRIES);
    SLOT_OFFSET =k*R_SLOT_SIZE;
  end
```

CSMA/CA algorithm for a transmitter on the RS485 LAN.

```
BACKOFF_INIT     = 20;
W_RSP_TIMEOUT    =1 millisecond;
MAX_TX_TRIES     = 20;
MAX_IDLE_TRIES   = 50;
```

wf, and *wg* - functions which return a maximum backoff number based on the input parameter.

Wait for a MAC_send call.

```
if p_flag is non-zero then
begin
  select a random number, i, between 0 and BACKOFF_INIT;
  SLOT_OFFSET =i·W_SLOT_SIZE;
end
else
  SLOT_OFFSET=0 ;

TX_RETRIES=0;
IDLE_RETRIES=0;

while TX_RETRIES < MAX_TX_TRIES and IDLE_RETRIES < MAX_IDLE_TRIES and not OK do
begin
  OK=False;
  detect an idle channel for SLOT_OFFSET+W_IDLE_TIME time units;
  SLOT_OFFSET=0;
  if channel is idle then
  begin
    send_frame;
    if a return priority response is expected then
    begin
      wait for response or W_RSP_TIMEOUT timeout;
      if a valid response has been received then
        OK=True;
      else (assume a collision has occurred)
      begin
        TX_RETRIES=TX_RETRIES+1;
        select a random number, j, between 0 and wf(TX_RETRIES);
        SLOT_OFFSET=j·W_SLOT_SIZE;
      end
    end
  end
  else (the channel is not idle)
  begin
    wait until the channel is idle;
    IDLE_RETRIES=IDLE_RETRIES+1;
    select a random number, k, between 0 and wg(IDLE_RETRIES);
    SLOT_OFFSET =k·W_SLOT_SIZE;
  end
```

Media Access Framing.

All frames must be preceded with a preamble which is necessary for bit and character synchronization.

SST RF frame preamble: 6 to 8 flag bytes (hex 7E).
SST RF frame trailer: 1 flag byte (hex 7E)
RS485 LAN frame preamble: 1 to 6 flag bytes (hex 7E).

The preamble is defined in addition to the flag start-of-frame delimiter. For example, frames on the RS485 LAN must begin with 2 to 7 flag bytes. The preamble is required for receiver synchronization.

Line Turnaround Timing.

SST RF turnaround timing.

A transmitting device on an RF link must be in receive mode and begin training less than 75 microseconds after shifting the last bit of the trailing flag byte of a unicast frame onto the link to guarantee that a response will not be lost.

A receiver on the RF link must wait at least 75 microseconds, after receiving the end-of-frame flag byte in a unicast command frame, before transmitting the last 5 bytes in the preamble of the response frame. Fast devices can begin transmitting sooner by adding extra flag byte(s) to the beginning of the response frame. Since each character, at 192 Kbps, occupies 41.7 microseconds, an 8-byte preamble is sufficient to remove the entire 75 microsecond wait time constraint. The first byte in the preamble must be sent within 200 microseconds.

RS485 LAN turnaround timing.

A transmitting device on the wired RS485 LAN must be in receive mode less than 75 microseconds after shifting the last bit of the end-of-frame flag byte onto the link in a unicast frame. Note that this time allows approximately 35 microseconds for transmitting a 16-bit abort sequence and 40 microseconds for the line turnaround.

A receiver on the wired backbone must wait at least 75 microseconds, after receiving the end-of-frame flag byte in a unicast frame, before transmitting the last byte in the preamble of the response frame. Since each character, at 460.8 Kbps, occupies 17.4 microseconds, a 5-byte preamble is sufficient to remove the entire 75 microsecond wait time constraint. The first byte in the preamble must be sent within 200 microseconds.